WHAT IS CLAIMED IS:

1. An analysis method, comprising:

identifying at least one construct in a program; and

interpreting, via an interpreter, a program on a processor, wherein during the interpretation, analysis code is invoked by the interpreter at the identified constructs, wherein the analysis code includes machine instructions for execution directly on the processor, and wherein the analysis code and the interpreter communicate via a predefined interface.

2. The method of Claim 1, wherein the construct is selected from the group comprising: a basic block, an instruction, a group of instructions, and a procedure.

3. The method of Claim 1, wherein during the identification of a construct, at least a portion of the trigger information is stored in a file for usage by the interpreter.

4. The method of Claim 1, wherein during the identification of a construct, at least a portion of the trigger information is stored is the binary program.

5. The method of Claim 1, wherein the predefined interface includes a registration procedure for the analysis code to register with the interpreter.

6. The method of Claim 1, wherein invoking the analysis code consists of providing to the analysis code at least one item selected from the group comprising: a null statement, a register value, a memory value, a program counter address, branch instructions, and an effective address.

7. The method of Claim 1, additionally comprising registering the analysis code with the interpreter via a predefined application programming interface.

8. The method of Claim 1, wherein interpretation comprises emulation.

9. The method of Claim 1, wherein interpretation comprises simulation.

10. A method, comprising:

storing a compiled analysis binary program, wherein the analysis binary program includes machine instructions from a first machine instruction set, and wherein the analysis binary program is configured to analyze or trace state information of an interpretable program; and

interpreting the interpretable program for execution on a processor,

wherein the interpretable program includes machine instructions from a second machine instruction set,

wherein the processor is configured to execute instructions from the first machine instruction set, and

wherein during the interpreting, upon encountering a selected construct in the interpretable program, the analysis binary program is invoked and is provided at least one item of state information about the execution of the interpretable program.

11. The method of Claim 10, wherein the state information includes register values, parameter values, instruction addresses, or data addresses.

12. The method of Claim 10, wherein the second machine instruction set includes generic machine instructions that are configured to be emulated on heterogeneous hardware platforms.

13. The method of Claim 10, wherein the construct comprises a procedure.

14. The method of Claim 10, wherein the construct comprises an instruction.

15. The method of Claim 10, wherein the interpretable program is a binary program configured for direct execution on a second processor.

16. The method of Claim 10, additionally comprising:

storing locations in the interpretable program in a file; and

using the file during the interpretation so as to identify the selected constructs.

17. The method of Claim 10, additionally comprising inserting a trigger instruction proximate to the selected construct, and wherein an interpreter is configured to recognize the trigger instruction as an instruction to invoke the analysis binary program.

18. The method of Claim 17, wherein the inserted trigger instruction is a machine instruction from the second machine instruction set.

19. The method of 17, wherein the inserted trigger instruction is a machine instruction that does not substantially affect the performance of the interpretable program.

20. The method of 17, wherein the inserted trigger instruction is a no-op machine instruction.

21. The method of Claim 10, wherein interpretation comprises emulation.

22. The method of Claim 10, wherein interpretation comprises simulation.

23. The method of Claim 10, additionally comprising ignoring selected machine interactions in the interpretable program.

24. An analysis or testing system, comprising:

means for storing an analysis binary code , wherein the analysis binary code includes machine instructions from a first machine instruction set, and wherein the binary code is configured to analyze or trace state information of an interpretable program; and

means for interpreting the interpretable program for execution on a processor, wherein the program includes machine instructions from a second machine instruction set,

wherein the processor is configured to execute instructions from the first machine instruction set, and

wherein during the interpreting, upon encountering a selected construct in the program, the binary code is invoked and is provided at least one item of state information about the execution of the program.

25. The system of Claim 24, wherein the second machine instruction set includes generic machine instructions that configured to be capable of being emulated on heterogeneous hardware platforms.

26. The system of Claim 24, additionally comprising:

means for storing user-identified locations in the program in a file; and

means for using the file during the interpretation of the instructions so as to identify the selected instructions.

27. The system of Claim 24, additionally comprising inserting a trigger instruction proximate to the selected instruction, and wherein an interpreter is configured to recognize the trigger instruction as an instruction to invoke the binary code.

28. The method of Claim 27, wherein the inserted trigger instruction is a machine instruction from the second machine instruction set.

29. The method of Claim 27, wherein interpretation comprises emulation.

30. The method of Claim 27, wherein interpretation comprises simulation.

31. An analysis method, comprising:

interpreting a binary program on a processor, wherein during the interpretation, analysis code is invoked by an interpreter, wherein the analysis code includes machine instructions for processing directly on the processor, and wherein the analysis code has been compiled prior to the execution of the interpreter.

32. The method of Claim 31, wherein invoking the analysis code comprises providing to the analysis code at least one item selected from the group comprising: a register value, a memory value, a program counter address, branch instructions, and an effective address.

33. The method of Claim 31, additionally comprising registering the analysis code with the interpreter via a predefined application programming interface.

34. The method of Claim 33, additionally comprising identifying at least one trigger using the predefined application programming interface, wherein encountering the trigger during interpretation causes the analysis code to be invoked.

35. The method of Claim 31, wherein interpretation comprises emulation.

36. The method of Claim 31, wherein interpretation comprises simulation.

37. An interpreter comprising:

code for interpreting a binary program on a processor, wherein during the interpretation, analysis code is invoked by an interpreter via a predefined interface, wherein the analysis code includes machine instructions for processing directly on the processor.

38. The interpreter of Claim 37, additionally comprising a predefined application programming interface that is defined by the interpreter so as to allow the analysis code to register and to define one or more callback routines.

39. The method of Claim 37, wherein interpretation comprises emulation.

40. The method of Claim 37, wherein interpretation comprises simulation.

41. A method, comprising:

compiling a source code so as to create a binary code, wherein the binary code includes machine instructions from a first machine instruction set, and wherein the binary code is configured to analyze or trace state information of a program;

modifying the program to include trigger information that identifies at least one trigger; and

subsequent to compiling the source code, interpreting the program for execution on a processor, wherein the program includes machine instructions having a second machine

instruction set, wherein the processor is configured to execute instructions from the first machine instruction set, wherein during the interpreting, upon encountering a selected instruction in the program that is identified by the trigger information, the binary code is invoked and is provided at least one item of state information about the execution of the program proximate to the execution of the selected instruction.

42. The method of Claim 41, additionally comprising selectively disabling certain of the triggers in the trigger information.

43. A method for dynamically translating instructions in original code written for a first architecture into native code for a second architecture, the method comprising the steps of:

designating at least one trigger in the original code;

translating the instructions into native code instructions; and

upon triggering the trigger, transmitting state information via a predefined interface, to analysis code.

44. The method of Claim 43, additionally comprising customizing the analysis code for operation with the original code.

45. An interpreter, comprising:

at least one interpreter module for interpreting a binary program, wherein the module provides at least one interface for allowing an analysis module to identify to the interpreter module trace information that is to be gathered during the execution of the binary program.

46. A method of instrumenting a binary program, comprising:

identifying at least one trigger location in the binary program;

storing the identified trigger location in a file that is separate from the binary program;

interpreting the binary program; and

invoking analysis code at the identified triggers.

47. A method of instrumenting a binary program comprising:

identifying at least one trigger location in the binary program;

storing the identified trigger location in a data section of the binary program

interpreting the binary program; and

invoking analysis code at the identified triggers.